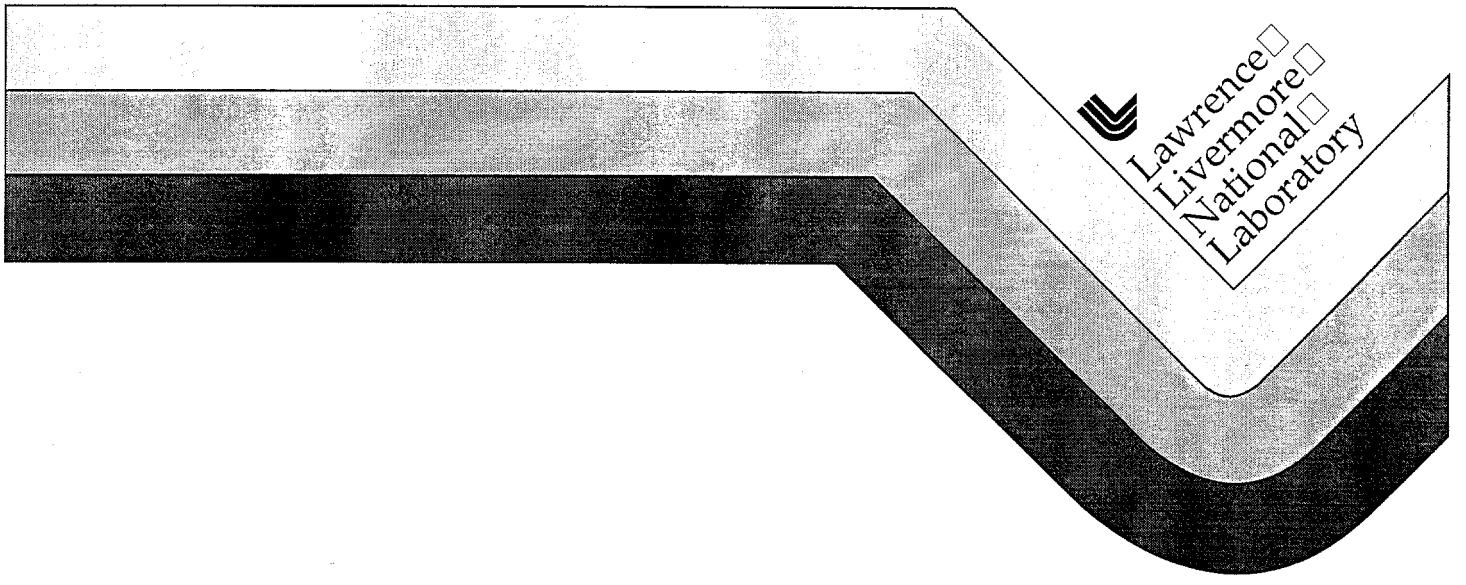
UCRL-CR-132131
S/C-B345870

National INFOSEC Technical Baseline Multi-Level Secure Systems

James P. Anderson Co.

September 28, 1998



DISCLAIMER

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

Work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract W-7405-ENG-48.

UCRL-CR-132131

National INFOSEC Technical Baseline

Multi-Level Secure Systems

James P. Anderson Co.

With the support of Lawrence Livermore National Laboratory
and the INFOSEC Research Council

September 28, 1998

UCRL CR 132131
URL: xxx

National INFOSEC Technical Baseline

Multi-Level Secure Systems

James P. Anderson Co.

With the support of Lawrence Livermore National Laboratory
and the INFOSEC Research Council

DRAFT

September 28, 1998

ACKNOWLEDGEMENTS

The author thanks Mr. Doug Mansur, Dr. Cynthia Irvine Mr. Charles Sherupski , Dr. Susan Gragg, and Dr. T.M.P. Lee for their thoughtful comments on an early draft of this paper. The logistics services of Mr. Douglas Price are also gratefully acknowledged.

MANAGEMENT SUMMARY

This report is a baseline description of the state of multilevel processors/processing prepared for the INFOSEC Research Council and at their discretion to the R&D community at large.

This report review the highlights of the development of Multilevel secure (MLS) systems, identifies the requirements for Mulitlevel secure systems, identifies outstanding issues affecting the availability of Multilevel secure systems and finally recommends an R&D program that addresses the issues and lays a foundation for developments into the 21st century.

Section 1 is an overview of what MLS is and contrasts it with several other currently popular terms; Multiple Secure Levels, Defense in Depth, and High Assurance computing.

Section 2 provides background, covering roughly three periods; early history (1964-1988), middle history (1988-1996) and recent events.

Section 3 gives a brief review of where we are today in terms of products, installations and Development capability.

Section 4 reviews MLS security requirements plus the use requirements that have become so important in recent times.

Section 5 identifies outstanding issues; everything from a need to focus on protecting classified information, the lack of systems development teams, the virtual unavailability of high assurance platforms for multilevel secure systems development, problems with the lack of label standards, and the inability to meet any end-user needs with what is available.

In Section 6, we outline development and research areas to address the issues raised in Section 5.

A brief summary of interviews with senior officers of the intelligence and defense communities is given in section 7 followed by a bibliography.

MLS Baseline Report

1. INTRODUCTION

1.1. Purpose of the Report

The purpose of this report is to provide a baseline description of the state of multilevel processors/processing to the INFOSEC Research Council and at their discretion to the R&D community at large. From the information in the report, it is hoped that the members of the IRC will be aware of gaps in MLS research. A primary purpose is to bring IRC and the research community members up to date on what is happening in the MLS arena.

1.2. Scope of the Review

The review will attempt to cover what MLS products are still available, and to identify companies who still offer MLS products. We have also attempted to identify requirements for MLS by interviewing senior officers of the Intelligence community as well as those elements of DoD and DOE who are or may be interested in procuring MLS products for various applications.

1.3. Outline of the Report

The balance of the report consists of the following sections; a background review of the highlights of the developments of MLS, a quick summary of where we are today in terms of products, installations, and companies who are still in the business of supplying MLS systems [or who are developing MLS systems], the requirements as expressed by senior members of the Intelligence community and DoD and DOE, issues and unmet R&D challenges surrounding MLS, and finally a set of recommended research topics.

1.4. Terms and Definitions

It is a little strange to have to start the review with definitions of terms, considering that MLS has been around for almost 30 years. However, of late there has been an erosion of precision of the terminology that could mislead the uninformed into believing that MLS was being developed, delivered and satisfactorily integrated into the national security fabric of this country. In this report, we will distinguish between MLS and several recent additions to our terminology, MSL and High Assurance Computing.

1.4.1. Multilevel Secure Systems (MLS)

By multilevel secure systems (MLS), we mean a class of systems containing information with different sensitivity levels (classifications, compartments) that simultaneously permits access by users with different security clearances and/or approvals, but adequately prevents users from obtaining access to information for which they lack authorization. Organizational objectives include control of access for disclosure and modification of data and the need for accountability of an individual's access to data.

MLS is also a mode of operation wherein all of the following statements are satisfied concerning the users who have direct or indirect access to the system, its peripherals, and products.
a) Some users do not have a valid security clearance for all the information in the system.
b) All users have the proper security clearance and formal access approval for that information to which they have access.
c) All users have a valid need-to-know (NTK) only for information to which they have access.
[implied: d. The system's operating system controls enforce the access and data movement implications of the statements above.]

There is an implication in the MLS mode of operation that the system has in place controls that ensure that the definitions work. In the early work, it was rapidly concluded that the operating systems needed to be in separate (from the users programs) protection domains, and of course to provide process isolation for those processes being run under them. Because the operating system was making security decisions about access and data movement, the notion of labels associated with various information objects was a key to providing the operating system with information about what data it was handling on behalf of the user.

MLS development was built on a security control concept based on the notion that all programs are ultimately run on behalf of some user; and all references by any program to any [information] objects need to be validated against a list of authorized types of reference based on the security authorizations of the user on whose behalf the program is running. This control concept was called the Reference Monitor [Schell 72] , and it contained three principles that were the foundation of its security:

1. The reference monitor must be tamper-proof
2. The reference monitor must always be invoked (mediates *every* access).
3. The reference monitor must be small enough to be analyzed and tested to assure that it is correctly designed and implemented. [Anderson 72]

A conceptual view of the Reference Monitor is shown in Figure 1. The Reference Monitor intercepts *all* Reads and Writes, and permits or denies them based on the authorizations contained in a per-user data base. The authorizations are an encoding of a security policy determined by some authority 'outside' of the model. If one imagines that the Reference Monitor has and maintains a representation of the user's current security level, the operation of the Reference Monitor is to compare the authorizations of the security level claimed by the user (and validated by the Reference Monitor) with those maintained in the policy database.

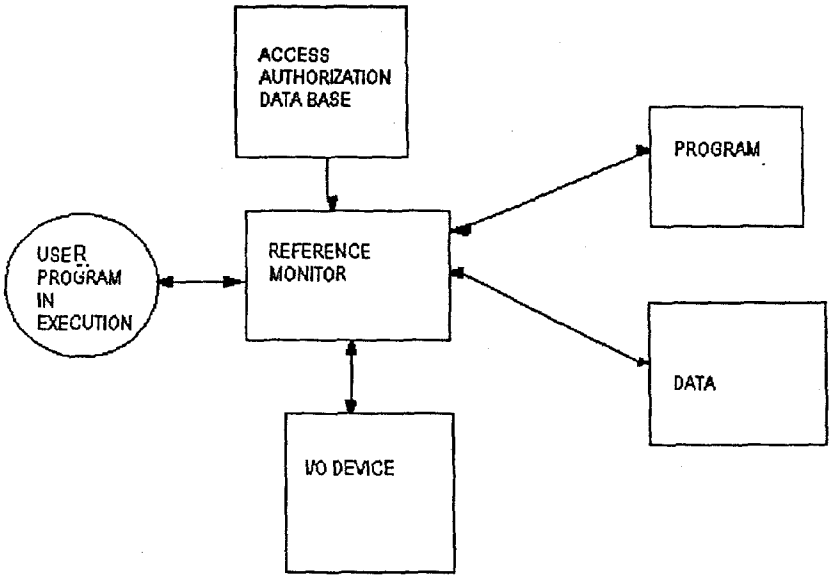


Figure 1
Reference Monitor

Originally, the implementation of the Reference Monitor concept was called a Security Kernel. An implementation of a security kernel meeting the Trusted Computer Security Evaluation Criteria (TCSEC) standard [TCSEC 83, 85] was called a Trusted Computer Base (TCB), and Systems incorporating TCB-based computers were called Trusted Systems. Because the TCSEC (the standard) allowed for systems that did access control based on access control lists (ACLs) only, there was a need to distinguish these from those systems that used electronic representations of classification and compartmentation labels that would be applied to a paper copy of an information object. This class of TCB-based systems was known as Multilevel Systems (or Multilevel Trusted Systems). Over the years, the MLS terminology has grown; sometimes to a bewildering degree. Depending on what kind of implementation one talks about, the terms involved are shown in Table 1 below.

MLS Terminology	Type of Implementation
Reference Monitor	Conceptual
Security Kernel	Implementation of Reference Monitor
Separation Kernels	Security Kernels for Process Isolation (only)
Trusted[1] Computer Base (TCB)	Implementation of Security Kernels per TCSEC Standard
Trusted (Computer) Systems	TCB-Based Systems[2]
Trusted (Computer) System	Trusted (Computer) Systems <B1
Multilevel (Computer) Systems MLS	Trusted (Computer) Systems >= B1, no Compartments
Multilevel (Computer) Systems MLS	Trusted (Computer) Systems >= B1 + Compartments
Multilevel[3] (Computer) Systems MLS	Trusted (Computer) Systems >=B1, only Compartments

Table 1. MLS Terminology (tree rooted in heaven)

[1] A computer or system is 'trusted' because it has been designed to a standard (the TCSEC) that if implemented correctly assures only secure (compromise free) operation, and the implementation is certified to meet the standard by an independent evaluation process.

[2] System is a Computer or a System and a Computer (i.e. network). For example, a File Server, or a Communications Server.

[3] Note that 'Multilevel' encompasses two or more adjacent hierarchical classifications, from the set U,C,S,TS, a hierarchy of classifications plus one or more compartments or categories or two or more categories.(since each category is treated as a 'level', even though they are technically disjoint (i.e., no category is greater than another).

Accompanying the MLS terminology are the 'modes' of operation of computer systems, that defined security-acceptable ways in which one could operate computer systems depending on what the content of the system was and the clearances of users with direct or indirect[4] access. It should be noted that these modes are still in use. Table 2 summarizes the modes of operation.

MODE	USER CLEARANCES	DATA CLASSIFICATION	TYPE OF POLICY	EXAMPLES OF IMPLEMENTATION
Dedicated	All cleared to all data	Single classification level or compartment	Physical access to the system = approval	Any kind of stand-alone system
System High	All cleared with[formal] need to know for all information in the system	Multiple classification levels and/or compartments	Discretionary	Any kind of stand-alone system with rudimentary controls.
Compartmented	All cleared and approved for *some* compartment(s) processed on the system. Not everyone approved for *some* compartments processed on the system. All cleared for all levels processed on the system.	Multiple compartments. (also multiple classification levels in most cases)	Discretionary	Any kind of stand-alone system with rudimentary controls. or Trusted system
Multilevel	All cleared for some information in the system (i.e. no uncleared users)	Multiple adjacent classification levels plus compartments or multiple compartments	Mandatory	Trusted system

Table 2. System Modes of Operation

1.4.2. Multiple Secure Levels Systems (MSL)

MSL is a term applied to the interconnection of enclaves[5], each operating at a System High level and (more or less) freely interconnected through Guards and security gateways. Because the higher level System High systems can interconnect freely with like-level systems and through Guards to lower level System High systems, the resultant network structure is called Multiple Secure Levels.

In practice, there are few security controls within an MSL enclave including those that might exist in the COTS software suite(s) provided within the enclave. The

[4] Receives product from a system, but isn't actively engaged in running programs *per se.*
[5] A distinctly bound entity enclosed in a larger entity, commonly thought of as LANs, and/or organizational-level networks.

operational model is that a user decides to share some data from within his enclave, and using his own best judgment, sends the data to a colleague in another enclave. In transfers from a 'high' level enclave to a 'low' level enclave, the Guard/gateway presents the document to a human operator who determines that the addressee is 'authorized', and who may scan the data for 'dirty words[6]' before passing it on. However, as long as the addressee is authorized, there is no real control of what can be passed between enclaves.

Note also that even with a human-in-the-loop Guard, that neither the human or the Guard will be able to detect even simple steganography[7], and certainly not the use of any sophisticated covert signaling channels. The use of automated guards will exacerbate the problem as sensitive information can be passed although no 'dirty words' are present. In such a case, the burden of judgment falls completely on the sender. With automated Guards, an attack program could pass sensitive information without the nominal involvement of a sender.

1.4.3. *High Assurance Computing*

In recent times, there have been projects carrying the label 'High Assurance' applied to computing platforms. The words are designed to give a 'feel good' glow to the listener, who is supposed to be lulled by the authority of the term. It is generally agreed that the term can only be applied in some context. High Assurance in terms of 'X'. For the purposes of this study, security, there is a plethora of possible values for 'X'; confidentiality, integrity, availability, self-protection, etc. We will use security as the example in the rest of the discussion even though we recognize that depending on the context, one could choose different values of 'X', and demonstrate that the object in question had the required assurance. We are better able to define how to achieve high assurance for security and integrity. Availability and other quality of service characteristics are more subjective.

The term is included in this report because it is encountered with increasing frequency, but rarely with a full definition of 'X'. High Assurance implies to some that the designer and implementor of some object have both "taken steps" to assure themselves that the object meets some design goals. For a given application 'Y', a high assurance system needs to provide with a high degree of certainty that the design requirements of 'Y' are met both by the design (intent) and implementation (practice).

Bill Shockley [Shockley 98] asserts High Assurance [for security] means at a minimum that the object and associated documentation must have the following four properties:

---

[6] Dirty-Words -a euphemism for compartment or special handling indicators on a document.

[7] Literally, 'hidden writing'. Basically, it is a technique to convey information by hiding it (varying the number of spaces between words, or hiding the message by using a designated letter of each word for the message, "Is Dean and Eddie thinking we allow spamming?? "(Read every 2nd letter)

I. Explicit Statement of Policies and Security (Confidentiality and Integrity) Functions Supported
Policy and execution model for access and data movement controls and audit trails. Careful functional specification of auxiliary policies supported; encryption, identification, authentication, signatures, etc.)

II. Soundness of the Specification, Architecture, Design and Implementation
Use of the Reference Monitor Concept, minimization (of the security relevant design), 'chain of abstraction', use of languages with well-defined semantics, and hardware mechanisms with well-defined protection semantics.

III. Specification and Design Actually Correspond to the Physical Object
This is configuration management, use of encryption for source/content integrity, etc.

IV. Credibility and Integrity of the People Involved
Only reliable people should be involved in the design/implementation of the High Assurance object. Clearing them would be good. Basically it says the object is only as good as your ability to trust the makers. [Experience in implementing High Assurance systems has shown that rigorous software engineering methods provide sufficient assurance that the system under development has not been subverted.]

In addition to these properties, it is observed that for really "High Assurance', that maintaining continuous assurance is implied. It does no good if the assurance effort is as indicated above, then the delivered object is just used, with no investment on the part of the owner in knowing more or less continuously its assurance state. This relates to configuration management - it starts at the requirements phase and continues until the object is retired from operational use. As one writer states "...it is expensive to create such a system. It is also expensive to own one. You can only play 'High assurance' if you are willing to engage with it. Once you start, you cannot stop... [Berson 98]

Since much of what we want to do with MLS requires High Assurance, it is useful to note that the MLS work of the past has included not only confidentiality, but system and data integrity, self-protection, etc. as an integral part of the design specification. It is what distinguishes MLS from various proposed methods of evaluation such as Common Criteria, ITSEC, etc. in that what is to be evaluated, contains specifications for what assurance it is supposed to have. That is, assurance can be assessed for two independent systems and the relative assurance of the systems compared.

1.4.4. Defense in Depth

This is yet another current phrase meant to convey how one is to protect networked systems. This phrase has some substance in that it is merely the common sense observation that one does not put all his security eggs in one basket, but rather

employs appropriate mechanisms to increase an adversary's risk of detection, penetration time and work factor, giving a defender time to detect a problem and adjust defenses and deploy countermeasures.

One briefing from the Unified Cryptologic Architecture [UCA 97] shows the layers of defense to include: transmission protection (encryption),

Firewalls (domain definition),
Intrusion detection,
Trusted Computer Base,
Applications security (built on TCB foundation),
I&A,
Access Control,
Audit and Misuse detection,
and Data Storage encryption/decryption.

Clearly, the combination of all of these elements achieves the objectives of increasing adversarial risk, penetration time and work factor. Unfortunately, defense in depth has also been characterized as lining up several 'low assurance' mechanisms and claiming that the sum total is equivalent to "high assurance" protection. This is similar to an argument sometimes advanced that 'any security measures are better than nothing' when all that is in place is a single mechanism (e.g. I&A),that gives full unfettered access to a complete system, and which provides no finer grained access or data movement controls.

2. BACKGROUND

2.1. Early History

2.1.1. In the Begining

In large measure, Computer Security as a discipline of Computer Science arose from the problems raised by the introduction and success of time-sharing. One of the earliest developments of time-sharing occured at MIT which developed a 7094-based Compatible Time Sharing System (CTSS) [Corbato 62, Crisman 65]. The CTSS was able to support approximately 30 apparently simultaneous users and was a demonstration of the concepts. CTSS shared only hardware among users. There was no sharing of programs or data that became the hallmark of later developments. Each user's program was swapped out in its entirety onto a high speed drum when the user's time quanta expired.

During the period of approximately 1960-1965, there were large strides in computer architecture, with multiprogramming and multiprocessor systems achieving maturity [Anderson 62, Davis 60, Barton 61, Iliffe 68]. Even IBM announced and eventually built a multi-processor configuration of the 360 system [Lett 67].

Not only was there an explosive interest in computer architecture, but the period saw equally spectacular growth in languages (stimulated by the Algol 60 report) and operating systems (interacting with multiprogrammed and multiprocessor architectures). All of this activity focused on the sharing of hardware, programs and data, and created the environment in which the computer security problems became prominent. The principal computer security concern of this time, reflected in the published literature, was raised by financial auditors and others over the changing environment for auditors to do their job, and the ability of the new electronic calculators to be instruments of fraud [Allen 60, Adelson 65, Wasserman 68, Schweishimer 70, Neville 71].

2.1.2. Multics

It is no accident that early academic interest in computer security centered at MIT. It was MIT that pioneered the techniques of time-sharing with the CTSS and it was at MIT that the information utility concepts were developed, that drove much of the design of Multics [Corbato 65, Organick 72]. Multics was the centerpiece of an ambitious project to create an 'information utility'. MIT had the experience of CTSS, but even so , its concepts of information sharing and the implications thereof were substantially addressed only for the first time in the Multics project.

It is perhaps only natural that an 'information utility' project would focus on information protection (i.e. protection of its resources) as a significant design objective [Glaser

65a,65b]. The design focus was on mechanisms that would permit controlled sharing of data in a utility setting.

It is difficult to overrate the contribution of Project MAC and the Multics design work to the development of information security concepts and computer mechanisms. The information utility concept brought many considerations of program and data sharing together such that it is hard to see how they would have been addressed in any other setting. It is also significant that almost 35 years later, Multics is still a model of what a secure time-sharing system should be! The current proliferation of many of its concepts in modern computers; descriptors, rings, segmentation and paging testify to the seminal nature of its development.

Multics contributed two concepts that remain important in the development of trusted computer systems today. The first is the use of descriptors as an address space management tool to represent a user's process. A descriptor is a generalization of indirect addressing that pointed to a segment containing program or data.

The descriptors contained not only a base and bounds address setting, but it also provided information about the type of segment to which it was pointing; whether it was data, program, file or a segment of other descriptors and what the user's privileges were with respect to the particular segment.

In Multics, the notion of 'rings' was introduced to provide a number of different hardware-enforced execution privilege levels in the system. At the lowest level, ring 0 (maximum privilege), was the core of the operating system; the memory management and process creation primitives of the system. At higher level rings, various facilities were to be found that would be useful for running user-oriented subsystems, but which one did not want the users to manipulate. For example, a data management system running as a shared application by all users would be found at a higher level ring. The outermost ring (the least privileged) was reserved for ordinary users and their programs.

For code segments (code segment descriptors), ring bounds were included that in effect defined the privilege level that had to be in effect in order for the code to be executed. Rings generalized the concept of privilege states in the same way that a collection of descriptors generalized the concept of address space.

The descriptors in Multics were only accessible for manipulation from procedures at a lower ring level (greater privilege). The effect of this design feature was to 'hide' the descriptors from ordinary users who ran at the outermost privilege ring. Outer ring procedures were only able to 'call' (controlled enter) procedures that existed in inner rings. The privileges accorded the procedure were determined at the time of the call by ring settings associated with the procedure found in the procedure's code segment descriptor.

The principal effect of the rings is to be able to make the entire operating system, including the ring 0 code an intrinsic part of each process running on the system in a totally protected way.

By the mid-1960s there was considerable research interest in protecting time-sharing systems from misadventures of its users. The focus, except for the Multics experience noted above was primarily on protecting the operating system from its users. Multics, as part of the information utility concept, focused on mechanisms that controlled sharing of data. By this time, there were a number of vendors with products supporting timesharing and multiprogramming. Some, such as Burroughs B5000 and D825 had multi-processor architectures. Others were single processor architectures (GE 635, Philco 2000).

Also by the mid-1960s, the multiprogrammed systems available had become reliable/enticing enough to have entered the Government market. With them came a recognition of the security problems of sharing. The security problems had achieved sufficient visibility, that Willis Ware of the Rand Corporation organized a session devoted to computer security at the 1967 Spring Joint Computer Conference (SJCC)[8]. Of interest were the several papers given during that session, including an analysis of the threats by Ware [Ware 67], and a comprehensive paper on security measures taken by NSA at that time [Peters 67]. Peters paper was quite interesting in that it provided a catalog of procedures and mechanisms that were needed to provide some level of security in multiprogrammed computer systems. It is interesting to note that many of his ideas such as audit trails, and management involvement in the evaluation process carried forward and showed up in the later and more formal computer security work.

As a result of the SJCC session, ARPA initiated a study of the computer security problem which was later turned over to the Defense Science Board (an advisory board to the DoD). The study, headed by Willis Ware, produced a report [Ware 70] that was classified CONFIDENTIAL and which delayed getting the results into the hands of the general public until a declassified edition was produced in 1979.

The output of the study were R&D recommendations and identification of the scope of the computer security problem and sharpening the focus of the problem. The study was important as the first 'official' recognition of the problem. It had a significant role in sensitizing the upper levels of Government management to the risks and vulnerabilities of using computers to store and process sensitive data.

Following hard on the heels of the Ware study, the USAF sponsored a computer security technology planning study the scope of which was specified to be "...a comprehensive plan for research and development leading to the satisfaction of requirements for multi-user open computer systems which process various levels of

[8] Throughout the 1960s, there were two computer conferences held each year sponsored by the Association of Federated Information Processing Societies (AFIPS). The conferences were held each spring and fall, and were known by the initials, SJCC and FJCC.

classified and unclassified information simultaneously through terminals in both secure and insecure areas."[AND 72]

The report of the study panel defined a program of research and development, the objective of which was to provide a rigorous definition of what security meant in a computer system and apply it in real computer systems. It is interesting to note that the elements required for security had all been identified earlier. What had been missing and what the study panel's report provided was a sound approach for determining whether a system's design was complete from a security perspective and whether the methods that had been developed or proposed in the past would provide the desired security.

The study had a number of interesting facets. These included the identification of the 'malicious programmer-user threat', the notion of the Reference Monitor, the concept of developing an operational model of what security means and designing systems to operate securely once started in a secure state, and finally the notion of a security kernel as an implementation of the Reference Monitor.

Computer security requirements are a response to the malicious user threat. The idea of the *malicious programmer-user* arose from the penetration experiments and experiences of the mid to late 1960s, as a generalization of penetrations and the threat of penetration. Essentially it was a short-hand for the idea that whatever was developed for security for computers, had to work even though one or more users of a system were malicious programmer-users, intent on stealing data or attacking the system itself. The malicious programmer-user was a metaphor for what is now called a technical attack; code inserted into systems the purpose of which is to steal data from those systems and export it in some fashion to outside of the system, or code inserted to cause a computer to fail or operate improperly at some crucial time. It was felt at the time that the malicious programmer-user concept was a powerful statement of requirement; that systems had to work securely even in the face of (being under) active attack. This concept resulted in the development of the Reference Monitor as an abstract description or conops of how a system designed to defeat malicious user attacks would operate. It should be noted that it wasn't until later that the risk of malicious code being entered into a system from non-programmers was really put into focus.

To a large extent, the plan was executed as conceived. Modeling efforts at MITRE [Bell 73] and Case Western Reserve University [Walters 74] independently produced a mathematical representation of the DoD Security Policy as stated in DoD Directive 5200.28. The formality of the model permitted arguments that the DoD policy formed a partially ordered lattice and that the model was general enough to cover all of the known security policies at that time[9].

[9] It is observed that the arguments about the applicability of the model are still going on today, with the adherents of the model saying in effect 'good enough' [Bell 88], and the detractors saying 'not mathematically sound'[McLean 87]. The most recent discussion of the topic was held at the 1996 IEEE Conference in Oakland [Shockley, and Blakely].

Shortly after it publication, MITRE produced a proof-of-concept 'security kernel' for the PDP-11, built according to the precepts outlined in the ESD study. [Schiller 73, Lipner 74]

Through the 1970s, engineering approaches to security of large systems used in military applications were proposed. These included the Job-Stream Separator [Schacht 75] as a control computer acting as a front-end to a large system that would be operated in periods processing mode. This concept was never built, but provided an interesting approach to doing secure processing on unsecured/unsecurable equipment.

Work was also undertaken to retrofit Multics [Whitmore, et. al. 74] to remove those design aspects of the system that would prevent the demonstration of its security soundness, and to shrink the Ring 0 (most privileged) code to just that needed to support security.

2.1.3. Grand Schemes

In 1974 Stanford Research Institute (SRI) proposed to develop a "Provably Secure Operating System" (PSOS)[Neumann 74]. The proposed R&D attempted to combine the focus on security (with the formal model of Bell & LaPadula) and the on-going work in program verification (formal proofs of properties of programs not related to security at that time).

The PSOS project ran for about 4 years. While it did come up with an operating system; one that was provably 'secure', the level of security proven was discretionary. PSOS and a contemporaneous project undertaken by Systems Development Corporation (SDC) produced competing formal specification methods. The SRI effort was called Hierarchical Design Methodology (HDM), and the SDC effort was called Formal Development Methodology (FDM). In spite of large sums of money, and some demonstration level proofs, the efforts have not caught on as a means of designing systems (secure or otherwise). As will be discussed below, cost-effective means of providing assurance is one of the key ingredients missing in computer security work.

Other highlights from the 1970s and into the 1980s were AUTODIN II, an attempt to design and implement a multilevel secure message switch, and KSOS (Kernelized Secure Operating System), an attempt to build a 'UNIX compatible' secure operating system. It was intended that the project run on two different platforms; PDP-11 and HIS 716. The HIS portion of the project eventually became SCOMP for the HIS716 (with no guarantees about UNIX 'compatibility'). When Wang bought the HISI SCOMP, it became the Wang XTS300 and is today the only B3 rated trusted system.

2.1.4. DoD Computer Security Initiatives and the Formation of the Computer Security Center

Starting in 1977, the DoD in collaboration with the NBS[10] Institute for Computer Science and Technology, began to sponsor a series of meetings called the DoD Computer Security Initiative. The impetus for these meetings came from the ASDC3I, the objective of which was "...to achieve widespread availability of "trusted" ADP systems for use within the DoD [Walker 80]". In addition to holding annual meetings, the initiative provided a forum and basis for lobbying to create a recognized technology center and program of/for computer security at the national level.

After some debate as to where the new entity should be placed in the Government structure, the National Computer Security Center was created in January 1981 with the Director, National Security Agency as the executive agent. The first Director of the Center was Melville Kline, a career NSA officer. His deputy was Col. Roger R. Schell, USAF. Together they built the center from 3 employees when they started to on the order of 250 professionals at its peak.

2.1.5. Publication of the Orange Book

The first year of the Center was focused on getting itself established. However, one the first Center Project/Products was initiated shortly after its formation; the standardization of what means 'secure'. The work on the standard actually started 1978 at an NBS conference in Orlando which resulted in a conference report that contained a definitive paper on providing criteria for evaluation of computer security[Lee 78]. As an outgrowth of that report, MITRE, in support of the DoD Computer Security Initiative produced a set of proposed computer security evaluation criteria [Nibaldi 79a, 79b, Trotter 80].

The standard, titled " Trusted Computer Security Evaluation Criteria" was published by the Center in August 1983. It became an instant best seller and to this day is better known by its descriptive cover color, the Orange Book(OB). The criteria were revised in late 1983 to incorporate some minor word changes. It was reissued as DoD standard 5200.28-STD in 1985, retaining its title and familiar cover.

Shortly after publication of the Orange Book, the Center published "Environmental Guidelines for Using the DoD Trusted Computer System Evaluation Criteria [Brand 84] as CSC-STD-8x-003, more commonly known as the "Yellow Book". The guidance related the security ratings of the TCSEC to operational environments defined by the modes of operation from the DCID 1/16 [DCID cite]; Dedicated, System High, Compartmented and Multilevel. Tables giving the recommended *minimum* TCSEC rating for systems as a function of the highest clearance of the least cleared user and the system's maximum data sensitivity and a partial incorporation of physical and procedural security items are found in the book. Since then the Center has published 36 Guidelines and Interpretations of the Orange Book.

[10] Now the National Institutes of Technology

Through the mid-late 1980s and into the 1990s, Trusted products began to appear. Perhaps impelled by the DIA-sponsored Compartmented Mode Workstation, a large number of B1 rated systems that meet the CMW requirements also began to show up.The principal requirement of the CMW was to automatically keep track of the security level of a document (window) as it is constructed by an intelligence analyst. It also supported cut-and-paste between windows. By the early 1990s, there were 14 trusted systems products (nearly all Unix based) rated at B1 or better; 5 of which were B2 or better.

2.2. Middle History [1988-1996]

Around 1988, there was a massive push by NSA to provide security through the MISSI program. MISSI stood for Multilevel Information Systems Security Initiative, and was essentially a bid to see how much security could be implemented using cryptography as the foundation. It is believed to have started with the objective of having a TCB running on each desktop/end-user's system. The program that was touted at the time was one of using cryptography to protect data in transit between enclaves. More importantly, the program envisioned the use of trusted systems (MLS systems) at workstations and servers. To this end, NSA commissioned Trusted Information Systems (TIS) to build a trusted version of the Carnegie Mellon MACH operating system. The project was known as TMACH.

From the start, the TMACH project was suspect. Individuals inside TIS indicated early on that it would not be possible to get even a B2 rating for the TMACH effort; that demonstrating some of its security properties would be impossible. After about 6 years of work on the project, NSA terminated the effort, declaring in effect, that building a High Assurance secure operating system was 'too hard'. Whether it is or not in an absolute sense is open to discussion, however, for the TIS/NSA group the answer is clearly that it was.

The demise of TMACH approximately coincided with the decision by NSA X to not build an SCI-approved version of Fortezza[11]; Fortezza +. The ostensible reason was that there was no particular demand for the device. However, it is believed that the decision was in part derived from the fact that they would have no High Assurance operating system to enforce Red/Black separation and label-based key and algorithm selection needed at that security level. As a result, the X Division, inheritor of many of the NCSC personnel and much of their charter, backed off from High Assurance trusted systems and continued to 'market' the notion of System High enclaves protected by Guards and Gateways as adequate for the 'Secret and Below'

[11] Fortezza is a crypto device in a PCMCIA card designed to support 'callable' crypto; that is a high-grade crypto algorithm available to application programs and the like that can be 'called' much as a subroutine is called to perform its function. The Fortezza was designed especially for the Defense Message System (a re-do of military communications to capitalize on Internet technology). It is revolutionary in the sense that it is a departure from the in-line 'black box' cryptography of the past.

environment which they believe makes up the bulk of the customer base for their products.

There are virtually no solutions being proffered for connections between TS(/SCI) to Secret, or between TS/SCI entities or between Secret to Unclassified entities. Many Commands having a need for TS-S connections are forced to use man-in-the-middle Guards, or in some cases, where the communications are totally stereotyped (formatted messages), an unattended Guard is permitted. The security effectiveness of Guards in this situation is totally dependent on the ability of the Guard (or its attendant) to recognize inappropriate traffic. Where the Guard acts as a Firewall, checking that traffic is addressed to pre-authorized recipients, the risk can be mitigated somewhat, but cannot in general be eliminated.

2.3. Recent History

Although it was started earlier (ca. 1993), the DoD published in 1996 a series of Information Systems Architecture documents that defined a Defense Goal Security Architecture (DGSA). This architecture is a blueprint aimed at interoperability and reduction of costs achieved by creating and maintaining special versions of applications that are commonplace in the commercial environment. TheDoD DGSA is an ambitious and far-reaching program, that if implemented would standardize applications across the board, thus achieving interoperability between organizations and among the armed services.

The DGSA has a security model that it is claimed is derived from a need to have multiple active policies in/on a platform. The model defines a *domain* as a policy and the objects of that policy. In the model, domains are independent, and there is no hierarchies within or between domains. The access control rules of the DGSA model require an accessor to be a member of a domain in order to read an object. For each domain, the 'user' (domain owner? Policy setter? Person 'responsible' for the data) is supposed to determine 'sufficient' protection for his data based on the value of the data and the perceived threat.

The DGSA presents a type-domain model of computing with requirements for which there are no 'worked examples'. Following a strict interpretation of the DGSA model, representing the DoD mandatory security policy over the four classification levels would require a TS-cleared individual to be a member of the U, C, S, TS domains with transfer privileges from U to C, U to S, U to TS, C to S, C to TS and S to TS. Realistically, for some users there is also requirements for write permissions of TS to S, TS to C, TS to U, S to C, S to U, and C to U. If one has to make a distinction between U and Sensitive but Unclassified(SBU) one can add an additional 8 permissions to allow a TS user bi-directional transfer authority.

Of course, all the permissions need to be managed for each user. When one adds domain membership and read-write and execute and transfer privileges associated

with them, the administration problem quickly becomes very complicated. To cap it all, since the security problem is about data and resource sharing, (re?)introducing the concept of need-to-know and user-managed ACLs on data expands the domain notion beyond simple comprehension. It appears that one would need a domain for each individual with whom he shares data for whatever reason. Further, it appears that the domains are on a per-application basis, so in order to do intra-organizational e-mail, one would have to create an e-mail domain and be a member of each other user's e-mail domain with whom one wants to correspond. If in addition, I want to share a draft paper on some subject with a colleague it appears that I have to have transfer privileges from the domain in which the draft paper is created (a Microsoft Word domain?) to my own e-mail domain. Then, to send the e-mail, I have to belong to the e-mail domain of the recipient and have transfer privileges to that domain. What would have to happen if the colleague with whom I wanted to share the draft paper for review had not been a part of any domain with me, is most unclear.

In the example above, what happens if the recipient is not a member of my Microsoft Word domain, have I not just done a 'downgrade'?

Two years after the initial publication of the DGSA, there has yet to be designed or built a proof of concept for the idea. The most recent paper that describes a set of 'challenges' to the OS design and implementation community [Feustel and Mayfield 97] makes claims of how much 'easier' a DGSA implementation would be compared to an equally comprehensive MLS implementation. However, without an actual system built to specification of the DGSA, such assertions are merely that. A recently released Institute of Defense Analysis report from November of 1997 gives a more formal definition of the DGSA model, and attempts to show how an implementation could have been put up on TMACH[12] [Schneider, et. al. 97]. The report mentions, but does not describe how the DGSA would usefully implement an MLS.

[12] Trusted Mach, a defunct product of Trusted Information Systems

3. WHERE WE ARE TODAY

3.1. Products

The Evaluated Products List (EPL) maintained by NSA and containing the names and vendors of trusted products at all (C1,C2,B1,B2,B3,A1) levels of certification, show two A1 systems only as network components, one B3 system (the Wang XTS 300), and four B2 systems all of which are no longer being offered by the designated vendors. In addition just entering the evaluation process at the B2 level is Data General's DG/UX. There are 7 corporations offering B1 operating system products, of which 3 are large mainframes and four smaller (workstation) systems. The smaller systems are almost all CMWs. However, since the assurance level of the CMW/B1 is so low, it is not considered further in this note [Lee92].

3.2. Acceptance (Installations)

It is hard to identify all the installations of CMWs and Trusted Systems. There are an undetermined number of CMWs in use in various segments of the Government; mostly DoD. However, CMWs are *not* considered very secure by serious practitioners because of their only B1 rating. However, they have done well because they are virtually the only label-processing systems available. even if the label processing is an applique on an essentially C2 base. This is not to say that B1 systems are useless; only that they are not considered to have sufficiently high assurance to protect TS data, or even Secret data in an environment where the risk range is Secret-Unclassified. CMWs are mentioned only because they do provide a form of label processing, and may be expected to interact with higher assurance systems in TS-Secret connections. The main problems reported with CMWs is maintenance, and COTs available for general use.

Unfortunately, the number of high assurance systems supporting MLS are few in number. WANG Corporation, who has the only B3 rated system on the market, has been relatively successful, having on the order of hundreds of their systems in use, or on order *as Guards*. In a recent meeting with them, they indicated that they hope to sell thousands of the XTS300 as Guards to the NII market in support of the Defense Message System. There has been virtually no interest in the XTS300 as a B3 system capable of MLS operation.

3.2.1. Assurance

Up until the mid-90s, systems were evaluated under the Orange Book standard for trusted systems. Starting circa 1992, pressure was applied on the U.S. to join in a European promoted effort to generate a common criteria against which all systems could be evaluated for security properties. There was an approximately one-year effort on the part of the U.S. to prepare their version of such a criteria to present as

part of the negotiation package when the criteria would be hammered out. After approximately 3-4 meetings of an approximately 20 people, a draft criteria was pushed through. It, like the European version was very heavy on process, with virtually no technical standards or requirements. All evaluations are to be done to a 'security target', basically a per-system statement of what it purports to offer in the way of security, with complete descriptions of the features, functions and mechanisms involved and how they are implemented. The security targets are created by vendors to describe what they think their customers need, and by customers as a statement of what they desire. There is no requirement that there be an overarching policy which the targets must address, nor that a particular target must have a (sub) set of known functions and mechanisms.

While the Common Criteria (CC) were being hammered out, the European community's strawman criteria, known as the ITSEC, was developed, and several systems were evaluated against its process. It might be worth noting that the TCSEC had worked examples which were used as the basis for the C to A classes. The developers of the Orange Book knew a priori that systems at each evaluation class could, in fact, be built.

It is understood that the U.S. is adopting the CC as its basis for evaluating the security of systems. It is hoped that by adopting the CC, that efforts of U.S. companies will be accepted by European customers, and that the total market for security products will be broadened accordingly. Since the CC is in the final throes of acceptance, there has been no evaluation under the CC per se. Rather, the ITSEC, which is for all practical purposes the CC, has been used as an interim basis for a few evaluations.

CC	TCSEC	ITSEC
EAL0	D	E0
EAL1		
EAL2	C1	E1
EAL3	C2	E2
EAL4	B1	E3
EAL5	B2	E4
EAL6	B3	E5
EAL7	A1	E6

Table 3
Evaluations Correspondence

Table 3 shows the correspondence between the evaluation levels for each of the criteria. EALx in the Common Criteria stands for 'Evaluation Assurance Level'.

For the CC evaluation classes corresponding roughly to the Orange Book classes C1, C2, B1, evaluations are 'farmed out' to organizations who have been certified by

the NIST as being competent to perform the evaluations. For the higher OB classes, it is expected that the NSA in the form of one of its evaluation groups will perform the work, possibly assisted by other organizations (MITRE, Aerospace) as they are today. The hope is that the evaluation process will be more rapid, and timely under the new procedures. What is not discussed is what standards are to be used against which the evaluations will be done.

3.3. Centers of Competence (Development Centers)

Approximately 10 years ago (ca. 1988), there were on the order of 10 companies in the business of developing multilevel systems and or applications. Today, one would be lucky to find 3 [WANG, Trusted Information Systems, Trusted Computer Systems, come to mind] companies capable of undertaking the development of a multilevel application. Even so,of the two or three companies who could undertake such a development, it would be with mostly inexperienced personnel, since most of those who may have done such development in the past have been dispersed, and are working in other fields altogether, or exist as a single point of competence in an area that needs several.

There are virtually no university or college programs where MLS issues are the focus of the program. The exceptions arethe University of Utah which is pursuing a program of operating system architecture research that is funded by NSA and DARPA. Their current focus is on supporting multiple policies in a system. The Navy Postgraduate School at Monterey is working MLS issues as part of their course work on OS in general. With these exceptions, present university programs are focused on various variants of intrusion detection. Considering everything, one cannot fault the universities, since they propose what they perceive their customers (Government in general) tell them they need.

4. REQUIREMENTS

4.1. General

4.1.1. MLS Security Requirements

4.1.1.1. A System of Security

The MLS security requirements are drawn from what is needed to implement a *system of security* in computer systems and networks. A system of security encompasses all of the elements that make up or contribute to the protection of data in a computer system or network. In particular, we include the fundamental notions of operating system security (security kernels, reference monitor) as a key set of components of the system of security. Because we are also attempting to deal with networks, some of the requirements have to do with protection of information while it traverses communications over which the user has little or no control. While we do not discuss in this report physical, procedural, personnel, or technical security, they too are as much a part of a system of security as any of the elements discussed below.

4.1.1.2. Elements of a System of Security

We define the term *security access class* (sometimes called security 'level') to mean the combination of a hierarchical classification (U, C, S and TS) and non-hierarchical, independent categories (e.g. WHIZBANG, U.S. Only, compartments, projects, etc.) that when taken together, represent the sensitivity[13] of information.

One access class is said to *dominate* another if the classification of the first is greater than or equal to the classification of the second, and the category set of the first completely includes the category set of the second. Where the classification of two elements of data are identical and the category sets are identical, technically, each dominates the other. The practical effect of such equi-dominance is that *either* element can initiate and respond to the other in situations where dominance of access class is necessary to effect access/connection/response/etc.

4.1.1.3. Computer and Network Requirements for a System of Security

The requirements for a system of security then are:

1. All users are uniquely identified to the computer or network system.
This requirement is one of several common sense requirements, but is really to establish accountability (and ownership).

[13] We use the term sensitivity to mean 'importance' in a broad way. It is meant to convey the notion that the sensitivity (represented by an access class) determines how much protection the data requires. Note that this is independent of perceived 'threat' and the so-called 'threat-based security.

2. Each user has a security access class (as defined above) associated with him or her.
This is a formality that says that users must have clearance (including *no clearance*) and access approvals for the categories. A user may *claim* a security access class *less* than that authorized to him in order to operate at give level; for example, UNCLASSIFIED, in order to assure that a report does not incorporate classified data by accident.

3. Each terminal /console/printer/communication link/peripheral device, etc. has a security access class.
This requirement is needed in many organizations where terminals may be in locations unsuited for some classifications or categories. One might also like to restrict some functions from some terminals. The functionality authorization is used to control what can or cannot be done from a specific console or terminal. The terminal/console/communications link/peripheral device may also have a functional authorization that in conjunction with the/a functional authorization associated with the user determines her ability to exercise some roles (from some terminal).

In networks, it is sometimes convenient to associate an access class or access class range with a particular port, in order to control the flow of data through that port, and assure its proper labeling.

4. Each data (or other) object has a security access class.
This requirement arises from the fact that proper handling and protection of data in the security system is based on the ability to compare the security access class associated with data (the object) with the security access class of the 'process' (job, session) (subject) referring to them. The trick is to identify the proper size of a 'data object'. There are some obvious cases; files are natural data objects. Records are also natural data objects. Trusted systems developed to the TCSEC standard have in the past generally associated access classes with files.

5. A session (job, process) security access class is determined from 2 and 3 above.
The idea of a session having a security access class derived from the terminal (workstation) and user security levels and which controls all references to data and marks new information created within the domain of the session (process) is fundamental to a system of security. The session/process is the surrogate for the user, and as such is subject to the same kinds of controls the user is subject to.

This kind of approach is suitable in systems where users interact with static data files through programs that terminate when the user terminates his session.

In a control environment, where a control process associated with a long term event, such as a flight may be continuously running for a long period of time with different users attached to it at different times, one could think of the control process itself as the protected object (instead of just the data), and require that people

attaching to it have a security access class that permits them to do so. Then authorized users could exercise the functions of the control process, read, input and otherwise manipulate the data of the control process depending on their authorizations.

6. All references[14] to any data objects will be allowed or denied based on the security access class of the data object and the security access class of the process/job/session making the reference.
The security access of the process making the reference must be greater than or equal to the security access class of the object being referred to. (i.e., the referencing process must *dominate* the object being referred to).

This requirement states that access to data (or other objects) is controlled by the security access class of the session/job/process making the reference. The reason for controlling access by the security access class of the session/process rather than the user *per se* is that by controlling access by the session/process assures that the only data objects that can exist in a session/process are all of a security class *dominated* by the security class of the session or process.

The concept of 'Need-to-Know' is provided by operating system design and a philosophy of protection that prohibits access to data or programs unless they are explicitly made accessible by the owner.The owner may authorize access to individuals or groups by name by putting them in an *access control list* (ACL) associated with the data or program object. Any process with a security access class greater than or equal to that of an object and running on behalf of an individual named on the object's ACL can have the kind of access specified for the name.

7. Transmission of a data object from a session/process or writing to another user/session/process will be governed by the rule that the security access class of the transmitting session/process is less than or equal to the security access class of the receiver.
This requirement is necessary to control information flow; that is, to prevent the inadvertent or deliberate transmission of data from a 'higher' security access class environment (process) to a 'lower' security access class environment (process).

Note that this restriction does not prevent users with individual security access classes greater than a particular process (e.g. another user) to communicate with it (him). What it does mean is that the 'higher' cleared sender has to claim (for the purpose of effecting the transmission)) a security level low enough to match or be included in the security access class of the intended receiver. It also does not prevent 'downgrading' of data as a controlled process.

---

[14] Typically, Read, Write (incldiuing Read), Append. It could also include Create, Destory

8. 'New' data objects created by a session/process/job are given the security access class of the creating process.
This requirement is needed for consistency. In general, the access class at which an object is created can be controlled by the user claiming an access class for his process that is appropriate for the desired classification, and is less than or equal to the maximum security access class the system will permit the user to invoke. Strict adherence to this requirement will constrain what the users can do with data. This is, after all, not bad, since the whole idea of a classification system is to control the access to and *flow* of information. This requirement also implies a requirement to be able to downgrade data.

To these more or less obvious and common sense requirements, we add those derived from the reference monitor concept;

9. The system of security must always be invoked.
What we have today is just discretionary controls. Discretionary controls rely on each officer's individual appreciation of what the security problem is, and how it is to be handled for each piece of data which he or she works with. Discretionary controls can be abused by malicious software.

10. The system must be totally self-protecting.
By self-protecting, we mean, as a minimum, that the operating system of the protected system cannot be induced, spoofed, tricked, etc. to give 'supervisor state' (e.g., root) privileges to any program on application. Rather than give supervisor state to a program, it should be designed such that it performs the privileged function(s) for the program in a controlled way. It further means that the operating system is able to isolate itself and its data from ordinary users While the system is generally not able to provide its own physical protection, this too is a requirement (to prevent attacks abetted by physical modification of the protection environment of the system. Self-protection is a very difficult requirement, especially when considering networks, since what is done in <u>any</u> part of a network (i.e., a system), may affect the network (system) as whole, such that one could inadvertently (and with the best intentions) weaken a node in a network to the detriment of the network as a whole. This area is probably still a hard research problem for the coming decade, but even without a total solution, it is possible to improve on present practice.

11. The security components of a system must be small enough to be understood.
With so much security distributed among so many places, this may appear to be an exercise in futility. However, requiring that whatever is proposed as a security measure be accompanied by a description of what problem it purports to solve, and its impact on the (a) network as a whole will improve things further, giving analysts specifics about which they can determine whether the claims are or even can be met. Whether it is possible to impose the discipline required to really deal with this problem remains to be seen. However, much effort is required to understand what is being done and why. This requirement comes directly from the

Reference Monitor notion, and is a charge to designers to keep it small for evaluation purposes.

12. There must be a way to manage the security.
Up to now, security management has focused on an individual who maintains the list of authorized users for a system, perhaps creating initial passwords, and allocating file space for a user. With systems now interconnected, the security management needs somehow to deal with individuals not under their direct control who have a need to interact with 'local' users an data bases on a system. Not only is this remote user not under local control, the local security management may not even know their counterparts in the remote user's organization.

There is an embracing of the X509 certificates as a key element in managing security of distributed users and distributed resources that they need to access. How X509 certificates will control access and authorization within a single organization let alone in the Defense community has yet to be articulated. Note that the Orange Book and its offspring were written well before these approaches were considered.

13. There must be a common security implementation policy among different participating organizations.
As an example, one must have a common representation of security labels; not just common formats, but the values that go into different fields need to be the same. If the X509 certificate becomes the common means of representing an individual to a system, it should be handled the same regardless of where it is presented. There are numerous other areas that need standardization to achieve the full protection potential of MLS.

14. There must be logging and audit functions to detect misuse, errors, and anomalies of data movement between levels and compartments.
In general, the whole area of audit and intrusion detection needs to be rethought with respect to a multilevel system. It is not sufficient to perform the logging from the system being protected. One needs means to detect anomalies and attacks without having to rely on the system under attack to operate properly. (Most attacks are detected after the fact).

4.1.2. Use Requirements

The original MLS work focused on providing a secure platform on which secure applications could be built. Today, users expect to be able to use the latest and greatest office productivity programs such as Word, Excel, Access, WWW, etc. Thus the new focus is on being able to run more or less arbitrary applications on an MLS in a secure manner.

This is new for MLS systems. As noted above, unless the present interest in MLS includes use requirements, the effort will fail for the same reasons that the earlier Trusted Systems requirements failed; they did not provide applications the users

wanted or needed to use. The Use requirements are more easily stated than those for security. Basically the Use requirements are:

1. The MLS system must run with virtually any workstation platform.
By this, we mean any users workstations without regard for manufacturer, or other details of architecture.

2. The MLS system must run virtually any software (Operating System and Applications) that would run on the workstation platform without the MLS system.
This requirement has been the principle stumbling block to the use of trusted systems in the past. It is the crux of the problem.

3. The MLS system must support present and foreseeable network services and applications.
In particular, it must support the present-day WWW and reduce/eliminate the impact of malicious/hostile code. It must be adaptable to whatever transpires in distributed computing over the next 5-10 years. In support of these requirements, it must provide integrated callable crypto. The form of the crypto (hardware or software) is not an issue at this point. Both forms should be provided for.

4.2. Agency/Department Requirements

4.2.1. CIA

One is almost tempted to combine the NSA and CIA requirements, but that would be an error, because while they are similar, they are not identical. Both of them would like to be able to create Virtual Private Networks (VPNs) both within and with other organizations on demand, with membership decided by a VPN 'manager' who is an operations or subject matter specialist, rather than a network, or computer specialist.

It is expected that membership in the VPN will consist of Government employees, Contractors, and Civilians possibly including foreign nationals. Due to the nature of the work, it must be possible to have full communication with and among all of the individuals, but also be able to selectively restrict access and data movement among all the individuals (but not all equally). Thus a subset of the Government employees may have access to, and data movement privileges with respect to a subset of the data, documents, or communications for a particular project. The other Government employees may have access to and data movement privileges with respect to a larger subset of the project data, but might be excluded from access to the most restricted subset. Depending on the project, it may be necessary and desirable for Contractors supporting the project to have access to some of the larger subset of the project data. However, they may be severely restricted regarding with whom they may share the data. Finally, the civilians may have the ability to communicate with the Government (and possibly the Contractors as well) about some aspect of the project

in which they play a role. All of the data access and communications require maximum protection, as they will be traversing the Internet or other networks.

With all of this, rapid creation, reshaping and dissolution of the VPN supporting the communications administration of the VPN is a key aspect of the requirements. Secure creation and distribution of identification and public key certificates for all of the members of the VPN including the contractors and civilian members will be a challenge to the key management system that supports the VPN process. This in turn puts extraordinary demands on the assurance and security properties of the systems on which the VPN administration and operation are implemented. It is asserted that the objective of being able to perform all of the VPN functions in the manner envisioned by the operations staffs will not be possible with the MSL or even a High Assurance system alone. It will require the full capability of a label processing (i.e. MLS) system to provide the controls required for the secure operation of the VPN. As noted above, all of this will have to be undertaken in the context of more or less arbitrary COTS application software as the basis for creating, managing and dissolving the VPN/project.

To support the VPN concept, one might like to have system support controlled sharing of files and data bases as well as assure the correct selection of cryptography and key based on who are the designated recipients of a particular message or document. In all cases, one would like the system supporting VPN operations to reliably and securely select the algorithm and key that permits the most secure transmission of the document to all designated recipients. If doctrine permits, it may be necessary to employ two or more algorithms to fully distribute the message to all recipients. All of the decision process should be supported and supportable by the MLS system on which it is implemented. In general, one would want the ability to override the controls in emergencies and force transmission through in less than security-optimum ways when necessary.

4.2.2. NSA

NSA also has a need for controlled sharing of information, but their environment is more stable. They too need to be able to create VPNs in effect, but the VPN is of longer term duration. Here there is a requirement to protect U.S. interests and data, and to maintain strict data in both intra-agency and public networks. In the public networks, there may be a requirement for anonymity as well. Here the broad system requirements translate into providing meaningful and effective security controls on what data may be sent where, while at the same time providing a flexible framework for reconfiguring resources to apply to topical problems as required to support military or diplomatic initiatives.

While NSA may also have a requirement for using and managing certificates for identification and key management, it is expected to be less dynamic than that required for the CIA. The operational override capability established for CIA use may not be appropriate for the bulk of NSA operations. However, it is believed that full

control over the capability can be managed through functional authorizations in the systems supporting the organizations.

4.2.3. DoD (DISA)

DoD requirements are for technology solutions to reduce resources dedicated to "watching" data movement across domains. This is particularly important in foreign liaison agreements. The DoD claims to 'need' the ability to support multiple policies on a single platform, and to move easily between 'domains' each roughly corresponding to one of the policies.[DIS96b] Presently, there is no support on the primary platforms for supporting any policies, as they are almost all COTS-software based systems. The DGSA has a requirement to prevent domain violations (i.e., data belonging to one 'domain' on a platform does not get mixed with data belonging to another domain on the same platform or by unauthorized movement of data from one domain to another on the same or different platforms.

For same-platform protection, a 'separation kernel' consisting of a set of security services is specified by the DGSA. COTS applications and operating systems will have to be modified to call the security services at appropriate places. However, the prospect of having to maintain changes to applications and operating systems that may affect the use of the separation kernel has tempered the enthusiasm for adopting this model, and to the best of the author's knowledge, there is no separation kernels being used for this purpose. The practical alternative is that enclaves are run at some System High. Thus as an approximation to the desired ability, the military commands utilize Guards to make sure that through error or malicious software, a domain violation does not occur when transferring data between enclaves.

Since there is no guarantee (i.e. there is no assurance) that a domain violation cannot occur on a platform supporting multiple domains, the Guards end up having to be attended by a human-in-the-loop, who is charged with visually inspecting a proposed inter-platform transmission and blocking any that appear to violate the domain separation policy being enforced by the enclave.

In order to successfully implement a domain separation policy, there is a premium on attendants who understand the full range of operations/activities supported by the enclave, and who are able to make informed judgments about proposed transmissions. There is already anecdotal evidence, that the human review is suspended whenever a situation arises where the traffic volume rises dramatically.

In a more prosaic venue, it is observed that capital ships (Aircraft Carriers, Cruisers, etc.) are increasingly dependent on intra-ship LANs for internal communication and operational support. Because the ships are frequently involved in highest levels of international diplomacy, or projecting power to remote parts of the world, they are supported by highly sensitive intelligence and operational resources, as well as required to store and act on highly classified operations plans.

The sensitivity of the data being moved about the ship is such that it is not possible to place it on a single LAN that connects not only the combat center of the ship with the command functions, but the cooks and bakers with the supply department, or the yeomen in the ship's office with the chaplain's assistants in dealing with emergency leave requests. Today, there are three or more LANs supported only because of the classification/sensitivity issues surrounding the data being moved on such LANs. This is not to say that one would not have redundant LANs for reliability or availability purposes. What one would like is a LAN network configured for maximum availability in battle, but with sufficient controls that one could easily mingle highly classified intra-ship traffic with routine, unclassified data that captures the rhythm of daily life on the vessel.

Here in a less likely setting is a model of a multilevel network, with the bulk of the traffic in and between nodes being unclassified, but with full flexibility for command or operations nodes to exchange data at the highest classification levels. As with other multilevel networks there is a requirement to support functional authorization as well as classification management. It would not do for a junior officer to be able to send an arbitrary message to the CINC of whatever fleet they are operating in without the Commanding Officer's approval.

It is hard to see how the enclave behind firewalls and guards will provide the operational and cost effectiveness required by fleet units, each of which is a network unto itself. While some might think that capital ships are a low threat environment(how are you going to get close enough to wiretap or intercept the intra-ship communications?), the fact is they are so large, and generally unprotectable, that placement of clandestine transmitters and/or wiretaps while they are in shipyards for overhaul or refitting is easier that anyone might imagine. Couple this with the fact that the crews on such ships may be as large as 3000-5000 men and women, the clearing of the entire crew is a virtual impossibility. Thus the 'low threat' environment picture for such vessels is largely illusory.

The bottom line for DoD applications is that in spite of clear examples where an MLS 'solution' had clear security and cost benefits, the pressure to utilize COTS 'solutions' without examining what that might mean, leads one to believe that the DoD is following a doctrine of expediency in order to conserve shrinking defense budgets.

4.2.4. Others

Of the remainder of Government entities, all but DOE can/could be served by solutions designed for the NSA/CIA/DoD applied to the specific instances of their problems. The DOE on the other hand adds one additional facet to the problem; very high speed computers, and networks to support them. The high speed aspect adds several specific items to the challenge; notably the array architectures of some of the very high speed systems, and the problems engendered by designing and building cryptographic services that can operate at the GH rates of modern high speed communications.

The DOE 'problem' (actually the High Performance Computing (HPC) environment problem) is characterized by having a number of HPC centers, linked by very high speed communications. The High Performance Distributed Computing model supported by such an arrangement is that sites with problems (not all of which are associated with nuclear weapons development), ship the code for their problem to one of the centers, with data as needed, and the center subdivides its array system(s) to provide a sub-array of a size suitable for the problem at hand. Due to flexible controls over the array systems, one could have one array processor in operation on two or more different problems. In effect, one has a two dimensional Job Stream Separator[Schact 75] problem (i.e., a Periods Processing problem) with the physical partitioning of the array of processors to be assigned to a problem. It is a characteristic of the array problems that once assigned, the problem code remains relatively static; generally there is no swapping or rapid overlays of code. Thus all the operating system needs to do is to keep the data input and output flows moving in as smooth a way as possible, and of course to not mix the input and output data between currently active problems.

The biggest challenge in this kind of architecture is be sure that the memory being used by a problem is completely erased after the problem is finished (either totally or in a quanta), before the processors and their local memory are reassigned to another problem. The other area, in which there is some experience, is making sure that the I/O among a number of problems is separated to avoid accidental or deliberate contamination of the data.

The research challenges in/of this kind of computing are being addressed primarily by DOE. [I have no details, but I understand that DOE is supporting an MLS operating system for array processors being developed by Intel and Hughes].

5. ISSUES

This section identifies issues and problems that will need to be overcome if there is any serious effort to be made to provide multi-level secure systems. The section could just as well have been labeled 'Problems'.

5.1. Need to Focus on Protection of Classified Data

For all the flurry about Infosec requirements in such documents as the DGSA, papers on Fluke, Flux, and new operating systems paradigms, there is precious little said about protecting (handling) *classified* data. It is probably no accident that proponents of using COTs security mechanisms for networked systems would prefer to talk about applications where the protection objectives do not bear the burden of classification, and where one can illustrate how protection of patient data in a hospital system can be easily achieved with the COTs mechanisms found in the various productivity suites, or how the financial community will benefit using cryptography to protect the transmission of financial transactions. These are good and even necessary applications. However, it is very hard to find any overt recognition that for the bulk of Defense-related applications, what is of concern is controlled sharing of classified data.

5.2. Lack of systems development teams

This is probably the single biggest problem facing the development of MLS systems. Fundamentally, it reflects a lack of commitment on the part of the Government in the pastto buying and using trusted computer products. The lack of commitment has spilled over into the research area, with only one program (U. of Utah) is devoted to any aspect of secure systems architecture, secure operating systems etc. With no developers to speak of, and no funding interest on the part of the Government, the situation is bleak. Attempts to get funding to demonstrate what might be done with trusted systems that meet the requirements outlined in section 4.1.2, have been singularly unsuccessful.

The problem will become even worse if the Infrastructure Commission<full name> recommendations are followed, and the funding to address those problems is forthcoming. What few people are available to work on MLS problems are surely going to be absorbed into Infrastructure work, especially if it is funded at a reasonably high level. What we need is one or two centers to play a catalytic role, such as MIT did in the 1960s and 1970s. Their charter should be directed to providing the foundations for trusted systems; re-examining old approaches, and comparing them against new ideas in this area.

5.3. MLS (Trusted Systems)[15] availability is virtually nil

As part of the effect of no commitment to MLS systems, the several companies who built high assurance MLS systems to the Orange Book specifications are no longer in business,or the parts that had the competence (e.g. DEC) have been disbanded. The single exception to this is the Wang Corporation. The B1/CMWs, are employed and deployed in parts of DoD, but they have insufficient assurance to support the full range of sensitivity levels that are called for in the various visions of how computing will be done in the Government in the 21st century.

The other problem with the old set of trusted systems is that they do not do current COTS software and are OS dependent. Here, what the users want is the latest and most 'modern' of programs or applications. The older trusted systems, based mostly on UNIX, are not equipped to run popular Office suites from any of the productivity vendors. We are thus in a dilemma, that the trusted products do not support the integrated productivity systems, and the COTS operating systems (Windows 95, 98) still apply security topically, with such features as passwords, audit trails, etc., but with no integration of the parts, and virtually no assurance of what is present..

While WANG would be delighted to have a monopoly on the trusted systems market, it has taken too long to exploit the B3 rating of its XTS300 system. Even today, WANG sees the 'big market' for the XTS300 as a guard in support of the DMS. For MLS to fill the needs projected, there will have to be other MLS systems available.

5.4. Interoperability

The principal interoperability problem that involves MLS systems is the system of labels used for classified information. If these are not the same everywhere, much time and effort will need to be expended in transforming them from one coding system to another. The general interoperability problem is hard enough without adding this to the list.

5.4.1. Label Standards

As has been pointed out repeatedly, labels associated with data objects and subjects are necessary because the whole concept of MLS is to have the operating system make access control and data movement decisions on behalf of the users. If it is unable to discern what kind of data it is encountering (i.e., no labels), it cannot do its job in that area.

Through 1990, there were approximately 12 studies[16] sponsored by the Government regarding security labels; what they should contain, and how they should be applied.

[15] See Table 1 for a terminology review

The most interesting finding of a survey of such studies was that virtually no two vendors used the same coding scheme for labels. Thus the simple representation of SECRET might be a number in one vendor's system, a letter in another vendor's system, an ASCII string in another vendor's system. In the case of the numeric values, one vendor might treat the highest numerical value as Top Secret, while another may have chosen '0' for TS. The point being that even with something as simple as the 4 classifications, there was no way to support direct comparison of labels from one vendor's system with those of another vendor.

The problem is re-emerging today with the advent of Certificates as the vehicle for carrying User ID, and/or Crypto keys. With the I&A use of certificates, one has to contend with different organizations representing clearances and authorizations in different incompatible ways. Ordinarily one might not care, but with so many of the Defense related agencies and departments becoming interconnected, and indeed making their data available to other interested parties in the Defense community, standard labeling is a must to prevent inadvertent mishandling of classified data.

There is no research *per se* required for this problem; rather, it requires a proposal for representing classification and compartments in certificates associated with users, and then using the same coding scheme for data labels. A pilot project implementing some/one of the certificate systems containing classification data would be useful in helping persuade and set the standards for all. The Intelink project is implementing certificates for I&A (primarily), and has as its data structure the following:
Nationality
Government/Contractor
Full common name
Organization (down to divisions)
Last 4 digits of SSN
Role
[garble]
Space reserved for an e-mail address
Space reserved for clearances

Note that the structure of the clearance/classification data has not yet been decided. Also, it isn't clear that this structure for certificates is going to be followed by other participants in Intelink..

5.4.2. Integrated Crypto

This is a desired capability that should be feasible if software crypto is involved. With an MLS system as a foundation, the cryptography could be assured that it wont be bypassed, nor compromised by applications running on behalf of the user. While it appears to be a stand-alone problem, it is clear that the promise of modern computing will not be met without a strong integrated cryptographic capability.

---

[18] [Gemini 86,Loscocco 88,Williams 88, SCC 89, Brinkley 89, Rogers 89, Abrams 90, Secureware 90, IHC 90, NCSC 90, Anderson 90, McCollum 90]

5.5. Assurance

Assurance is the hardest problem of building systems, especially MLS systems. It is still the field of the formal methods people, but in the 20 odd years of Trusted Systems, little has been done beyond what was current some 10-15 years ago. Assurance is and was the stumbling block over which the old program tripped. For example, it took almost 2 years to get a rating on a B1 system because of a combination of factors; vendor ignorance of what was required, lack of a successful worked example of the kinds of documentation required by the evaluators, criteria 'creep' (applying criteria from a higher level rating), no incentive for the evaluators to be more responsive. Higher level systems (B2, B3 and A1) could take 3 years or more.

If we expect to produce new MLS systems, waiting 2-3 years for an evaluation will put is back where we were 10-15 years ago. How can we get adequate assurance at the system level, at the applications level? Program proof hasn't done it, what else can we do? Process-heavy approaches such as NIAP and EC don't make it either in my opinion. The composition problem has stood in the way of building your own using certified parts.(Why? Has anyone ever tried to put together a multilevel OS using certified parts? Has anyone broken the multilevel problem into a set of necessary and sufficient parts (certified or not)? Whatever is done to alleviate the problem, it should be easier to apply than trying to write OS code without using the technique.

5.6. MLS solutions need to accommodate end-users needs

As was pointed out in section 4.1.2, MLS solutions need to be able to accommodate end-users needs. This has been *the* problem with the earlier effort at trusted systems. It is *the* problem facing anyone attempting to provide MLS solutions today. There are two possibilities for the short-term; a trusted file-server supporting single-level-at-a-time workstations running the arbitrary COTS operating and business systems. The second is to encapsulate the applications in VMs, and support them with a multilevel file system running in/on its own VM. Neither approach has been implemented. The VM approach is considered near to mid term development.

5.7. Need to identify impact on COTS applications (e.g., WORD, EXCEL, etc.) of adopting current proposals for security services to protect classified data.

The two most frequently heard proposals for providing security are to adopt:
a). ISO model for security 'services'

b) NT security mechanisms.
to protect multilevel classified data. This is a serious issue since there are various proposals, architectures or systems designs based on one or both of these elements. There are two objectives behind this issue:
First, to determine how widespread the changes would be to an application like WORD by adopting the OSI 'Services" suite as has been suggested in such proposals as the DGSA.

The second objective is to lay out explicitly how the mechanisms available in NT could be applied to support multiple data security levels in a single environment, and from this determine what the residual risk of compromise would be.

The result of this work will give a blueprint of what additional services, mechanisms, and assurance are required for protecting classified data.

6. RESEARCH RECOMMENDATIONS

6.1. Focus on protection of classified data

All of the recommendations in this section have this as a key component of the proposed work. The concept of controlled sharing of classified data in a multilevel environment underlies all of the work on Trusted Systems and their application. Thus the proposed projects in Section 6.x, are aimed at demonstrating protection of classified data. So too the work described in Section 6.2 and 6.x.

Even with the emphasis on classified data, the systems that provide this protection also provide a foundation upon which non-classified applications can be built, or operated. Thus the capability to represent categories or compartments allows an IT manager to segregate his users by major application such that accounts payable can only be accessed by individuals who are designated as belonging to the accounts payable category. When combined with an access and movement control audit, it provides the tools to deal with controlled sharing of virtually any kind of data.

6.2. Seed and nurture systems development groups

First, the current emphasis on COTs, while meeting short-term objectives for reducing (eliminating) stovepipe development costs is in the long term self-defeating. By emphasizing use of such products as NT, WIN9x, Microsoft Office, ti is inevitable that one needs internal experts to design and implement 'business' systems using these products. The internal employees become proficient with one or more of the products and are then lured into the commercial world at double or more their Government salaries. The point is that the COT-only solution has the seeds of its own destruction built-in.

The question that needs to be answered is what is the best strategy for getting OS and platform independent MLS systems into the market place? First, can the 'vendor-do-it-all' approach that was used with the early trusted systems work today.? While the reasons are many and varied, the biggest stumbling block to vendor participation is the perception that the Government misrepresented the potential government market for such systems, and did not buy even token amounts of trusted systems. The vendors remember this.

If that attitude on the part of vendors still prevails, then even if one wanted to move ahead on trusted systems, it will probably only be possible to contract for their design and implementation along with a fixed order for so many systems or identify a dual use market. . [Jelen 85] Note that this is not a particularly bad thing; it merely underlines how small the size of the market for trusted systems really is. Further, if the trusted systems are purchased under a contract to design and build them, the Government can easily require that the design and implementation be documented,

and the developer get the design and implementation rated at an appropriate level (at least B3). Thus the certification becomes a deliverable along with the hardware and software.

In addition to showing real interest in MLS development, there needs to be funding at one or more academic institutions that can play a catalytic role in Inforsec engineering comparable to that played by MIT in the 1960s and 1970s.

At the moment there are no institutions devoted to security systems engineering, although the US Navy Postgraduate School comes quite close. The academic quality is quite high, however, the student body is Navy and Marine Corps officers who are then returned to the fleet for general service.

One institution that has promise is the new Georgia Institute of Technology Information Security Center. They are just embarking on the formation of the Center, and have key academic and political support. If they are able to identify an MLS BHAG comparable to the Information Utility notion that energized the MIT Multics project in the 1960s, they will be well on their way There may be other candidates, but to make strides in Security Systems Engineering, one probably needs an Engineering based institution.

6.3. Demonstrate multilevel processing

This is recommended because there is no MLS system that operates with COTS applications, doing routine office functions.

6.3.1. MLS enclave with Trusted File Server

An early and relatively inexpensive demonstration could be to show multilevel processing on LANs with diskless workstations and a trusted file server.. The proposal is to place all the trust in a multilevel file server, and run associated workstations a single level at a time [Irvine 98]. An extension of the file server's TCB is included in each workstation to provide a trusted path from the user to the server TCB to establish the working security level for a session, and to erase all workstation storage upon change of security levels (i.e. enforce object reuse). The TCB extension may also be a local trusted repository for cryptographic support. (See Figure 2). Besides being a target through which end-user's needs will become articulated, such a project would show that something is happening; a jump-restart of an MLS program.

Until now, all attempts to provide MLS have focused on the use of trusted systems, operating alone or in networks, with each trusted system independently providing labeled MLS controls. The only systems of this sort commonly available today are the low assurance level B1 systems implementing the CMW. In virtually all cases these trusted systems fail to address the use/usage requirements posed above, although

they do provide rudimentary label processing. The software and applications run on the CMWs is the vendor's own word processing or spread sheet software. Since many of these systems are UNIX-based, the "productivity" software often turns out to be a UNIX application, or UNIX-compatible versions of standard software. These systems, while providing better security quality than ordinary COTS[17] do NOT run most of the applications of interest to the end users.

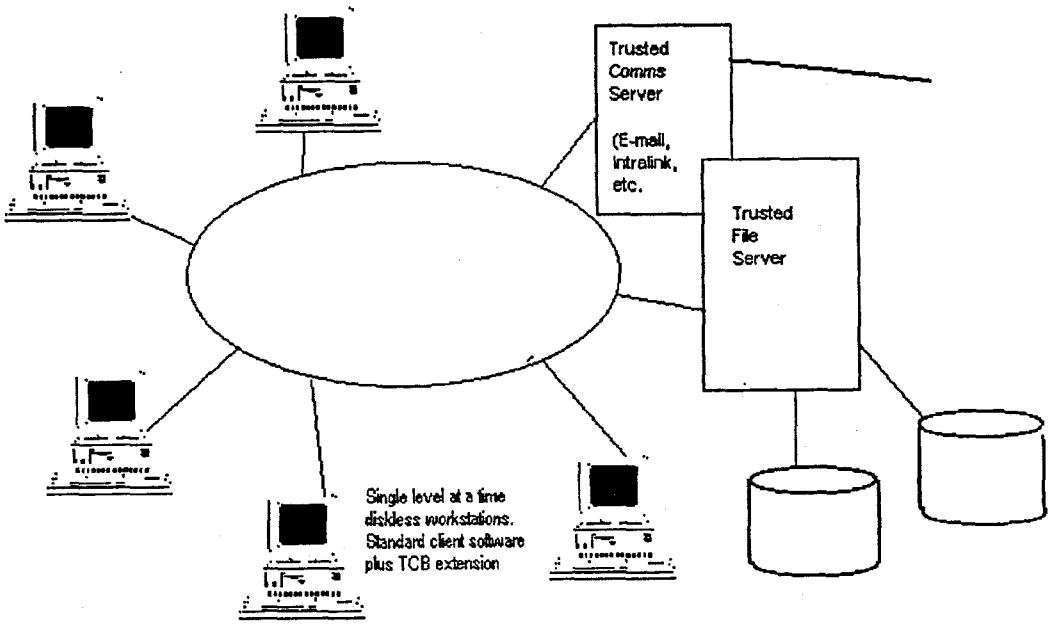


Figure 2
TFS as a Network Building Block

The TFS design is a MLS file server implemented on a high assurance trusted system. The MLS file server is the locus of security control in the LAN, mediating access to files and data bases in/on the file server by labels associated with users, sessions, and of course the data itself. The file server TCB (security kernel) has an extension that exists in the LAN diskless (writeless) workstations for the purpose of providing a trusted path (i.e. non-spoofable communications between the user and the TFS TCB), and to provide a reliable and trusted mechanism to control the purging of the PC memory at the end of a session.

[17] The 'B1' assurance level is the lowest possible using labels. It was never considered by the people doing the Criteria to be suitable for extensive ML applications. According to the Yellow Book [Brand 84], it is good enough mostly for adjacent clearances/classification (Confidential/Secret, TS(with an SBI) and one (a single) compartment, etc.)

6.3.2. Initiate an agressive review/project to make NT a B2 multilevel processing system

It is clear that NT is the applications development OS of choice these days by any number of developers. Using the NCSC evaluation of NT3.5 and the B Level Windows NT Feasibility Study sponsored by MISSI as the basis, actually implement the ideas.

This will require very good relations between Microsoft (MS) and the Government. However, what would appear to overcome that problem is the identification and selection of a 3rd party developer to create and maintain through all of the changes planned and made by MS a B2-level label processing version of NT. The strategy is to have a label processing variant of NT that will support a large number of security 'levels'; on the order of 16-32 hierarchical levels, and within and for each, 16K-64K compartments [This may even be a strategy of MISSI but we dont know that.]

It is recognized that there is risk in this approach, but it is believed that the risk will be significantly less than either doing nothing (what is presently happening), or hoping that some other company who will be more accommodating to the community and government's needs, will rise up and displace MS. The strategy (of buying security directly) also has the added attractiveness that it hasn't been tried before. The proposal is that the USG will fund the entire project, with the chosen vendor selected based on its knowledge of MLS and MS NT. One of the biggest risks is that MS will change NT in significant ways from one release to another. Often these changes are cosmetic 'churning', and could be ignored. However, the end result could be that the security community ends up using a version of NT that deviates further and further from the commercial version with each significant change or release.

6.3.3. Commission at least one other MLS system that 'does' NT

To feed MLS needs further downstream, I would recommend commissioning at least one other project to produce a mulit-level reasonably assured (B2 or better) stand-alone system based on NT (which should be the norm for all clients by the time we get all this done assuming it is even started) for laptops and notebooks. It is not all that is needed, but some of the rest is already underway.

Anothr area of investigation is whether the NT/CE for hand-held devices, toasters and the like is a candidate for providing a separation kernel, or the foundation for embedded (single function) systems that don't do labels such as routers, guards, dirty-word filters, or compliance systems (such as the SDNS Server that only serves to ensure that some kind of cryptography is employed on all outgoing transmissions). This could be the focus of a 6 month to 1 year study.

6.4. Interoperability

There are several facets of interoperability that need to be addressed, especially if it is intended that the result will be systems that provide protection for classified data.

6.4.1. Label Standards

In order to advance on this subject, it is necessary to first establish label standards for the Defense-related community. As noted in Section 5.4.1, there isn't much needed in the way of research as much as an agreement to what constitutes a minimum set of classification and distribution directives for data. These elements, in their propoer form then need to be incorporated into Public Key certificates. Because ir seems unlikely that all of the principals will adopt a common CA, the issue of cross certification and certificate management needs to be joined.

In some ways, the label standards issue is the easy part of the problem since it focuses on user identification. The second problem, representing authorizations and using the information to negotiate security 'associations' in connection with integrated cryptography has more unknowns.

6.4.2. Integrated Crypto

There are two components to the research; what are the preferred communications models for sessions, for E-mail and other file transfers, and what is preferred for make-break communications such as found in building Web pages, and the like. The communications models are important, since they will define where and how the protocols supporting these different modes of use will/must be integrated into systems. The focus of the work needs to be 3-5 years out when hopefully the political issues surrounding use of cryptography will have been resolved in one way or another. Even so, it will probably require some years of work to develop an integrated use model of systems communications, and from that what is needed to protect it. Also in the modeling, one needs to deal with the critical question of key management. It isn't clear whether the current emphasis on certificate management will fit all of the proposed uses of cryptography in the future.

As part of the research into integrated cryptography, a complete study of the key management 'problem' needs to be undertaken, with the first order of business being an identification of the issues that lead to the adoption of the X500 scheme of certificates, and what role they are supposed to play in networked use of systems. Along with identification of the issues, it would be immensely helpful to have a catalog of various key management approaches; everything from paper key lists to the various forms of electronic keying, their perceived scope of applicability and their strengths and weaknesses. The question of scalability of various proposed key management schemes needs to be addressed as well. One needs also to identify where high assurance is required in key management.

Of course, all of the emphasis on communications and cryptography has a bearing on the MLS problem. With software cryptography a economic reality, there is even more need to provide a high assurance MLS environment to provide security level separation and to assure that cryptography is properly applied.

6.5. Assurance

Until now, nearly all efforts to get assurance involve adopting an arcane philosophy of how to design programs about which one might be able to make some proofs. Two of the best known methods are SRI's Hierarchical Design Methodology (HDM), and the SDC effort called Formal Development Methodology (FDM), both brought to fruition during the early to mid 1970s, and applied to attempts to prove operating systems were secure. At some point, it became evident that one could prove anything about a program if one chose the axioms properly. In spite of large amounts of money and time spent, program proof is still an elusive goal, still without general acceptance by the programming community.

The problem appears to be that the formalisms associated with design methods are too far removed from an actual program and the translation of proven formalisms into machine code is rife with opportunities to make errors. The translation from formalisms to machine code represents a large semantic discontinuity. The problem is that there is no assurance that once the translation is made that the resulting code has any of the security properties proven for the abstraction.

What is needed is a less-formal method of representing security properties of code and a means of writing code to test at run-time that the security properties are still intact. This isn't quite the same as proving code is secure, but it may go far toward building assurance that code has not been changed in some way, and if simplicity is emphasized, could be a way to provide assurance about the global behavior of code. Whatever is considered here, it is important that the method be simple, easily applied, and consistent.

6.6. Systems architectures that support user requirements

6.6.1. Encapsulation Research

As a start, it is believed that the necessary concept for pushing forward with different/better architectures is to exploit the notion of encapsulation. The TFS approach mentioned in the development section above, puts all of the trust in the server, with only minimum extensions of the trusted elements to the workstations to provide a trusted path between the user and the server (to reliably establish an operating security level), and to assure the erasure of client memory (for object reuse) whenever the security level is changed.

The focus of this research should be on identifying those Multilevel applications that act as the inter-systems 'glue' for multilevel use. The obvious example that started the thinking in this direction is the File System; the heart of an operating system of the 1950-1980s. Note that the file system is different from a DBMS, but that the DBMS is yet another application for which MLS techniques were/are applicable and has resulted in trusted Oracle among others.

In our modern networked age, the idea of E-Mail as a multilevel application has emerged. Subsequent research has been identified that would attempt to determine whether the full range of network communications (i.e. all protocols, all applications (ftp, telnet, http, etc.) could be centralized in a single server, and if so, how to do so without having to design new programs from scratch). In a sense, the research is into multilevel communications management. [Not a bad title for some research proposals].

Regardless, the intent of this research is to identify as many multilevel applications as possible, and to identify a minimum multilevel framework to support them. One then would be interested in whether it would be possible to create distributed (process level) MLS systems composed of sets of the multilevel applications. The objective is to determine whether it is possible to create a set of simplified multilevel building blocks that could be assembled to produce two or more different multilevel systems. Of course, one would want to configure the systems dynamically in real time. The key underlying idea is that classified data would be handled properly by each component, according to the policy for handling classified data.

6.6.2. VMM Development

Other encapsulation strategies include the use of virtual machine monitors (VMMs) to project Virtual Machines(VMs). This clearly is a means to isolate individual users. However, it begs the question of sharing resources. In order to be effective, it is necessary to share files, data-base systems, etc. How this is accomplished efficiently in a VM environment, needs to be studied, and strategies for effecting the desired performance made known. For example one could assume that a trusted VMM existed for a machine powerful enough to support <TBD> users. What and how would the sharing as well as the separations be carried out? How does such an architecture deal with the multiple connections and distributed computing environment of today's web environment? The question of how one would implement Web technology in/on a secure VM systems could also be fruitfully explored.

6.6.3. Conduct research into sharing

With so many emerging architectures and computing models competing for our attention, and need to be ML Secure, it is necessary to identify for each what the sharing assumptions are and relate them to what we understand about sharing and its impact on security. In this light, what, if anything, can be done to protect sharing at the applications level. The so-called Data Driven Attacks (DDA) (also known as 'executable content') are the 'threat'. With so much effort being put into designing 'network' computers and similar systems, at least some part of the networks will have computationally challenged terminals with some or all of the programs they use coming from the network.

There are so far, only imperfect virus defense kinds of answers proposed as the means to defend against such attacks. Can a 'deferred commit' approach work(run in

an encapsulated environment such as a VM, and if an attack occurs, flush the application, otherwise, accept it)? Here the focus shifts from detecting the attacking program before running it to detecting the effect of the attacking programs. Even this may be insufficient, since the attack may not be to seize control of the target site, but to affect data in a shared data base. Depending on what the protection objectives are, secure networked computing may not be possible. Regardless, innovative novel ideas to deal with the problem should be sought. At least one will obtain a current baseline as to what is thought possible.

6.7. Identify impact on COTS applications of using available security mechanisms

The purpose of this project it to identify shortfalls (if any) of adopting COTS mechanisms or the ISO 'Services' approach to protecting classified information, and using this data to persuade vendors to augment their security mechanisms to cover the missing or less-than-desired parts.

To start, the project will require a description of the data environment, e.g., 'Documents, cables, spreadsheets, Web pages, etc. etc. existing at classifications running from Unclassified through TS/SCI.

Scenarios of common operations will be developed to include preparation or editing of simple reports, simple spread-sheet planning, vugraph preparation, collaborative work within and between Defense-related agencies, Special scenarios involving task-group activities with officers from several different Agencies will be developed to highlight the protection requirements and the interoperability aspects of what the scenario is all about. In all cases, the scenarios will focus on the need for protecting classified information.

In addition to the scenarios, a comprehensive statement of protection objectives will be derived from information security policy statements subscribed to by all members of the Defense-related community. The protection objectives may be accompanied by proprietary interest requirements that may add constraints or internal administrative steps to release some information/data to partner agencies.

Given the environment picture and a number of use scenarios, the project will be required to select from the ISO Services mode, or from independent COTS infosec mechanisms elements that will provide the classified information protection, and show how these elements, working together provide some or all the required data protection.

The project will then analyze the resultant structure to identify remaining vulnerabilities and from this assessment identify additional capabilities needed.

6.8. End Points

Within 5 years, have several different working models for using MLS principles in workplace environments. The proposed TFS and its offspring are only proposed examples that could be done in about 3 years. The objective of using COTS mainstream products at workstations is very much the whole objective. The more generalized solution the better.

Within 5 years, have a supported current (or at least no more than one Rev level behind) version of NT that does label processing and is at or near B2 level assurance. Risk: B2 may not be good enough for some applications. If that is so, then one may have to use other products that have higher assurance levels, or not do the application in the most convenient way. Security happens.

Within 1 year, determine if there are indeed viable alternatives to this strategy outlined above.

Within 2 years, produce one or more reports on the sharing assumptions/components of various network architectures, and for each a strategy to control the sharing. (no unplanned sharing). [This should be an on-going project, with reports issued every time a significant shift in architectural thinking takes place.]

7. APPENDIX I SUMMARY OF INTERVIEWS

During the preparation of this report, interviews were held with senior officers primarily of the Intelligence Community to attempt to get a sense of what is needed in terms of multilevel systems or multilevel processing. All were adamant about not being willing to adopt or use the trusted systems that had been built into the early 1990s. The reasons given were diffuse, but had to do with the systems not being able to run the applications of the moment, difficulty of maintenance, system inflexibility, and inability to run COTs applications. Some of the respondent Agencies had adopted a COTs work environment and cited as further evidence of their good judgement that the trusted systems of yore did not do Windows or Excel or Access, etc.

For a variety of good economic reasons, the Agencies are standardizing on various commodity software Operating Systems and productivity suites. The most common being Microsoft's NT and Office 9X . With Microsoft (and several other vendors) augmenting their products with Web software, the Agencies and Departments have in the suites all the software needed for Web operations as well as full productivity suites.

When questioned about how they expect to protect their sensitive data, the respondents became less specific. There is clearly a wish that Microsoft (or other vendors) would incorporate high assurance security in their products. They are quite willing to use whatever security mechanisms are provided in the productivity suites and show no particular interest in whether the mechanisms are part of a seamless protection scheme or not. If passwords are a feature, they will use passwords.

Nevertheless, they are concerned about their most sensitive data and want to control access to it and its movement. One of the officers stated his concern about the real danger incurred by interconnecting to and with so many different entities that the nature of their business requires. He is concerned that without strong controls, the legacy of sharing with partners of long standing may be in jeopardy. He stated further that any 'solution' that did not address the need to support more or less arbitrary COTS and GOTS programs would not succeed.

Yet another one of those interviewed expressed the opinion that MLS as it was understood is not relevant to today's problems but did not elaborate.

When asked specifically about the proposal to implement MLS on file servers, permitting operation a single level at a time, there was mild interest. However, because it has not been demonstrated, nor the cost of adopting such a model been worked out, there was no wild enthusiasm for the idea.

Unfortunately, in all of the interviews, there was no sense of how classified information was going to be protected, especially with all of the interconnectivity being

planned and implemented. Except for the nagging need-to-know issue, and the recognized weaknesses of enclave Guards, there is an acceptance of the enclave model as about the best they could do.

8. BIBLIOGRAPHY

Abrams 90 — Abrams, M., Eggers, K., La Padula, L., Olson, I, *"A Generalized Framework for Access Control: An Informal Description"*, Proceedings of 1990 National Computer Security Conference

Adelson 65 — Adelson, A.M., *"Embezzlement by Computer"*, Security World, September 1965

Allan 60 — Allan, J.A., *"Embezzlement by Electronic [Computer]"* Accountants Magazine, April, 1960

Anderson 62, — Anderson, J.P., et. al. *"D825, A Multiple-Computer System for Command and Control"*, Proceedings of Fall Joint Computer Conference, 1962

Anderson 72 — Anderson, J.P., *"Computer Security Technology Planning Study"*, Vol I and II, ESD-TR-73-51, October 1972.

Anderson 90 — Anderson, J.P., *"Overcoming a Limited Number of Categories in Application of COTS Trusted Computer Products"*, TN9005-001/242890, June 1990

Barton 61 — Barton, R.S., *"A New Approach to the Functional Design of a Digital Computer"*, Proceedings of the Western Joint Computer Conference, 1961

Bell 73 — Bell, D. E., and LaPadula, L.J., *"Secure Computer Systems, Mathematical Foundations,* "Vol I and II ESD TR-73-278, MITRE Corporation, November 1973.

Berson 98 — Berson, T.A. Private Communication

Brand 84 — Brand, S. *"Environmental Guidelines for Using the DoD Trusted Computer System Evaluation Criteria."*, Proceedings 7th DoD/NBS Computer Security Conference, September 1984.

Brinkley 89 — Brinkley, D., Badger, L., Rogers, K., *"Guidelines to Labeling in Trusted Systems"*, Final Report for NCSC, April 1989

Crissman 69 — Crissman, P. A. (Ed.) *"The Compatible Time Sharing System: A Programmers Guide"*, MIT Press 1969

Corbato 62 — Corbato, F. J., Merwin-daggett, M., Daley,R.C., *"An Experimental Time Sharing System"*, AFIPS Conference Proceedings, Vol 21, 1962 SJCC

Corbato 65 — Corbato, F.J., Vyssotsky, V. A., *"Introduction and Overview of the Multics System"*, AFIPS Conference Proceedings, FJCC 1965.

Davis 60, — Davis, G. M.,": *The English Electric KDF9 Computer System"*, The Computer Bulletin, No 4, Vol 3, British Computer Society, London 1960.

DCI xx — Director of Central Intelligence Directive 1/16, <date, etc.>

DISSA 96a — Defense Information Agency, Center for Standards, *"Department of Defense Architecture Framework for Information Management (TAFIM)"*,Version 3.0, April 1996.

DISSA 96b — Defense Information Agency, Center for Standards, *"Department of Defense (DoD) Goal Security Architecture (DGSA)"*, Version 3.0, April 1996. Volume 6 of TAFIM6

Gemini 86 " — *Categories, Compartments and Mandatory Security"*, Gemini Computers, Inc., July 1986.

Glaser 65a — Glaser,E.L., *"A Brief Description of Privacy Measures in the Multics Operating System"*, AFIPS Conference Proceedings, SJCC, 1965.

Glaser 65b — Glaser,E.L., Coleur, J.F., Oliver, G. A., *"System Design of a Computer for Time Sharing Applications"*, AFIPS Conference Proceedings, FJCC 1965.

IHC 90 — *"IHC Security Labeling Working Group Draft Preliminary Report"*, October 24, 1990

Iliffe 69 — Iliffe, J.K., *"Basic Machine Principles"*, American Elsvier, New York, 1968.

Irvine 98 — Irvine,C., Anderson, J. , Hackerson, J., Robb, D., *"High Assurance Multilevel Services for COTS Workstation Applications"* to appear in Proceedings, NISSC, October, 1998

Jelen 85 — Jelen, G. *"Information Security: An Elusive Goal"*, xxxx, 1985

Lee 78 — Lee, T.M.P.L., et.al., *"Processors, Operating Systems and Nearby Peripherals: A Concensus Report"*, in 1978 Conference on Audit and Evaluation of Computer Security II: System Vulnerabilities and Controls.", NBS Special Publication 500-57, April 1980.

Lee 92 — Lee, T.M.P.L., *"A Note on Compartmented Mode: to B2 or not B2"*, Proceedings of the 15th National Computer Security Conference, Oct 1992, Baltimore, MD.

Lett 68 — Lett, A.S., Konigsford, W. L.,, *"TSS-360: A Time Sharing Operating System"*, AFIPS Proceedings, FJCC 1968.

Lipner 74 — Lipner, S. B., *"A Minicomputer Security Control System"*, MITRE Corporation, MTP-151, Bedford, MA, February 1974.

Loscocco 88 — Loscocco, P. *"A Dynamic Network Labeling Scheme for a MLS LAN""*

McCollum 90 — McCollum, C., Messing, J., Notargiacomo, L., *"Beyond the Pale of Mac and DAC - Defining New Forms of Access Control "*, 1990 IEEE Conference on Security and Privacy

Neumann 75 — Neumann, P.G., et. al., *"A Provably Secure Operating System"*, SRI Technical Report, Project 2581, June 1975.

Neville 71 — Neville, H.G., *"Computer Capers Herald New Crime Wave of Embezzlement"*, The National Underwriter: Property Edition, August 1971

Organick 72 — Organick, E.I. *"The Multics System"*, MIT Press, Cambridge, MA, 1972.

Nibaldi 79a — Nibaldi, G. H., *"Proposed Technical Evaluation Criteria for Trusted Computer Systems"*, M79-225 MITRE Corporation, October, 1979.

Nibaldi 79b — Nibaldi, G. H., *"Specification of a Trusted Computing Base (TCB)"*, M79-228, MITRE Corporation, November 1979.

NCSC 90 — *"Guidelines for Labels in Trusted Computer Systems (DRAFT)"*, NCSC, 12 January 1990

Peters 67 — Peters, B. , *"Security Considerations in a Multi-Programmed Computer System"*, AFIPS Conference Proceedings, SJCC 1967

Rogers 89 — Rogers, K., Zuckerman, S., *"Sensitivity Labels - Expanding the Lattice"*, Unisys Defense Systems, 1989

SCC 89 — SDNS *Access Control Document* (Revision 1.3), July 1989

Schact 75 — Schact, J., *"Jobstream Separator System Design"*, MITRE Corporation, Bedford, MA

Schell 72 — Schell, Roger R., *"Notes on an Approach for Design of Secure Military ADP Systems,* presented at Privacy and Protection in Operating Systems," Proc. ACM Annual Conference, August 1972, published in Preliminary Notes on the Design of Secure Military Computer Systems, USAF Electronic Systems Division MCI-73-1, January 1973

Schiller 73 — Schiller, W. L., *"Design of a Security Kernel for the PDP-11/45"*, MITRE Corporation, MTR-2709, Bedford, MA June 1973

Schneider 97 — Schneider,E.A., Feustel,E.A., Ross, R.S., *"Assessing DoD Goal Security Architecture (DGSA) Support in Commercially Available Operating Systems and Hardware Platforms."*,Institute for Defense Analysis, P-3375, November, 1997

Schweisheimr 70 Schweisheimr, W., *"Embezzlement by Computer"*, Bankers Monthly, June 1970.

Secureware 90 Secureware, *"Trusted Interprocess Communications; Notes on SECUREWARE Specificiations for Trusted Interprocess Communications"*. Revision A, April 1990

Shockley 97 Shockley W.. Private Communication

TCSEC 85 Trusted Computer Security Evaluation Criteria, DoD Standard 5200.28-STD, NCSC 1985

Trotter 80 Trotter, E. T., Tasker, P.S., *"Industry Trusted Computer Systems Evaluation Process"*,MITRE Corporation, MTR3931, May 1980

UCA 97 Unified Cryptologic Architecture, Technical Architecutre, UCA2010 Study, National Security Agency, August 1997

Walker 80 Walker, S. T., *"DoD Computer Security Initiative"*, Proceedings of the Third Seminar on the DoD Computer Security Initiative Program", November 1980.

Walter 74 Walter, K. G., et. al., *"Primitive Models for Computer Security"*,ESD -TR-74-117, Case Western Reserve University, January 1974.

Ware 67 Ware, W.H., *"Security and Privacy in Computer Systems"*

Ware 70 Ware, W.H., *"Report on Defense Science Board Task Force on Computer Security, Security Controls for Computer Systems."*, R609-1 Office of the Secretary of Defense, Washington, D.C. (republished in 1979).

Wasserman 68 Wasserman, J.J., *"The Vanishing Trail"*, Bell Telephone Magazine, July 1968.

Whitmore 73 Whitmore, J.,Bensoussan, A., Green, P., Hunt, D., Kobziar, A.,Stern, J., *"Design for MULTICS Security Enhancements"*, ESD-TR-74-176, Electronic Systems Division, Air Force Systems Command, Hanscom Field, Bedford, MA 01731 (Dec. 1973).

Williams 88 Williams, J, Day, M, *"Sensitivity Labels and Security Profiles"*, Proceedings of IEEE Conference on Security and Privacy, 1988